# مسائل محلولة بلغة البرمجة باسكال

م. محمد العليان

@mhdalyan

**للتواصل مع الكاتب**

LinkedIn

about.me

Twitter

# مقدمة

يحتوي هذا الكتيب البسيط على حلول لبعض المسائل العامة والمحلولة بلغة البرمجة باسكال، تشمل المسائل مواضيع عديدة منها التعامل مع المجموعات والسجلات والملفات بنوعيها النصية والثنائية ، كما تحوي مسائل في المؤشرات بشكل معمق ومكثف من خلال مسألتين كبيرتين تغطي أفكار هامة وعميقة في عالم المؤشرات.

هذا الكتيب ليس موجه للأشخاص الذين يودون تعلم لغة البرمجة باسكال وحسب، وإنما لكل من يريد أن يبني فكر برمجي قوي ومتين.

يستعرض الكتيب مسائل في أبحاث المجموعات(set) والعمليات عليها، التسجيلات (Records) واستخداماتها، التعامل مع الملفات النصية والثنائية، مسائل ضخمة وعميقة جداً في المؤشرات والتي هي عصب البرمجة بالنسبة لنظام التشغيل، بالإضافة إلى مسائل متفرقة.

الشيفرات البرمجية تم تجريبها وهي تعمل بشكل صحيح، كما أنها متوفرة للتحميل من خلال منصة **Source forge**.

سأكون مسروراً حقاً بملاحظاتكم على هذا الكتيب، وأرجو ألا تبخلو بها.

آمل من الله تعالى أن يكون هذا الكُتيب مفيداً لكم وأن يقدم العون إلى كل من يريد أن يتعلم البرمجة عموماً ولغة الباسكال خصوصاً، وأرجو أن يكون عملي هذا في صحيفة أعمالي، والله من وراء القصد.

دمشق في 7-7-2014

محمد العليان

## المسألة 1 :

سوف نتعامل في هذه المسألة مع مجموعات قواسم الأعداد الصحيحة الموجبة لتساعدنا على إيجاد العديد من العلاقات الكائنة بينها.

سنقصر الاهتمام في البداية على مجال الأعداد الموجبة من 1 إلى 255، إذ يمكن ألا يتمكن الكثير من المترجمات التعامل مع مجموعات تتعدى هذا المجال، ولكن الحل سيبقى عاماً.

بداية سيقوم البرنامج بإيجاد مجموعة القواسم الخاصة بكل من أعداد المجال المدروس وسنخزن لكل عدد مجموعة قواسمه ماعدا نفسه وفق البنية التالية:

```
 Const MaxN=255;
Type range= 1..maxN;
Type SetDivizer= set of range;
Type TabSetDiv = array [range] of SetDivizer
```

نعرّف إذاً نمط جدول مجموعات أعداد صحيحة TabSetDiv، ونخزن في متحول من هذا النمط مجموعات القواسم. فإذا كان لدينا المتحول VTabSet من النمط TabSetDiv، عندها تكون نتيجة الخزن لمجموعات القواسم فيه كما يلي:

```
VTabSet[1] =[1];
VTabSet[2] =[1];
VTabSet[3] =[1];
VTabSet[4] =[1,2];
..............
VTabSet[20] =[1,2,4,5,10];
..............
VTabSet[51] =[1,3,17];
..............
VTabSet[255] =[1,3,5,15,17,51,85];
```

يُطلب من البرنامج حساب مجموعة القواسم الخاصة بكل من الأعداد في المجال 1 إلى 255، وخزنها في جدول وفق البنية المبينة آنفاً، ونقوم على التوازي بوضع مجموع قواسم هذه الأعداد في جدول، كما يطلب كتابة برنامج جزئي procedure يسمح بإظهار محتوى مجموعة أعداد صحيحة.

والآن يُطلب من البرنامج باستخدام المجموعات وجدول مجاميع القواسم، إيجاد مايلي:

أ-إظهار مجموعة قواسم أي عدد ضمن مجال المسألة.

ب-إظهار الأعداد الكاملة: (العدد الكامل يساوي مجموع قواسمه).

ج-إظهار مجموعة القواسم المشتركة لعددين.

د-إظهار مجموعة المضاعفات المشتركة لعددين والواقعة ضمن المجال.

هـ -إظهار مجموعة الأعداد الأولية.

الأكواد البرمجية لجميع مسائل هذا الكتيب موجودة على الرابط التالي :

http://sourceforge.net/projects/pascalgeneralproblem/

```
program set_text;
const maxn=255;
type range =1..maxn;
type setdivizere=set of range;
type tabsetdiv=array [range]of setdivizere;
type tabsum=array[range] of integer;
var vts,dts :tabsetdiv;
    seet:setdivizere;
    sum:tabsum;
    num1,num2,i,j,n:integer;
    c:char;
procedure creat_setdivizere(var vst:tabsetdiv;var sum:tabsum);
begin
  vst[1]:=[1] ;
  sum[1]:=1  ;
  for i:=2 to 255 do
   begin
     for j:=1 to i-1 do
      begin
         if (i mod j = 0) then
           begin
             vst[i]:=vst[i]+[j];
             sum[i]:=sum[i]+j;
           end;
      end;
   end;
end;

procedure creat_double(var dst:tabsetdiv);
begin
  for i:=1 to 255 do
   begin
     for j:=i to 255 do
      begin
         if (j mod i =0 )then
          dst[i]:=dst[i]+[j];
      end;
   end;
end;


procedure menu(var c:char);
begin
 repeat
   WRITELN('                          THE MENU                               ');
   writeln('                        (S)ET DIVIZER                            ');
```

```
    writeln('                      (C)OMPLETED NUMBER                         ');
    writeln('                      (D)IVIZER BETWEEN                          ');
    writeln('                    (M)ULTIPLYING BETWEEN                        ');
    writeln('                           (P)INARY                              ');
    writeln('                           (E)XIT                                ');
    WRITELN('*****************************************************');
    writeln('WRITE THE FIRST LETTER OF THE ORDER  YOU WANT TO DO');
    readln(c);
  until  c in ['s','S','c','C','d','D','m','M','p','P','e','E'];
    WRITELN;
end;

procedure read_number(var n :integer);
begin
  writeln('READ THE NUMBER YOU WANT');
  readln(n);
  while (n<1)or (n>255) do
   begin
    writeln('YOUR NUMBER IS WRONG.......... PLEASE ENTER IT AGAIN ');
    readln(n);
    WRITELN;
   end;
end;

procedure print_set(s:setdivizere);
begin
  writeln('THESE ARE THE ELEMENTS OF THE SET');
  WRITELN;
  WRITELN;
  for i:=1 to 255 do
   begin
     if (i in s) then
       write(i,' - ');
   end;
  writeln;
  writeln;
end;

procedure divi(n:integer);
begin
  print_set(vts[n]);
end;

procedure complet(var s:setdivizere);
begin
  for i:=2 to 255 do
   begin
     if (sum[i]=i) then  {if the number = sum of(divition) sum[i]; procedure}
       s:=s+[i];{add i ; complet number to set } {remember sum[i] type of setdiviser  }
   end;
end;
     if (sum[i]=i) then  {if the number = sum of(divition) sum[i]; procedure}
       s:=s+[i];{add i ; complet number to set } {remember sum[i] type of setdiviser  }
```

```
   end;
end;

procedure divbetween(num1,num2:integer;var s:setdivizere);
begin
  s:=vts[num1]*vts[num2];
end;

procedure doublebetween(num1,num2:integer; var s:setdivizere);
begin
  s:=dts[num2]*dts[num1];
end;

procedure pinary(var s:setdivizere);
var pin:boolean;
begin
    s:=[2];
  for i:=3 to 255 do
   begin
      pin:=true;
      j:=2;
      while (pin)and(j<i) do
       begin
         if(i mod j =0)then
          pin:=false;
        j:=j+1;
       end;
   if(pin =true)then
    s:=s+[i];
   end;
end;
/////////////////////////////////////////////////////////////////
Begin { program Start }
 seet:=[];
 for i:=1 to n do
  begin
    vts[i]:=[];
    sum[i]:=0;
    dts[i]:=[];
  end;
  creat_setdivizere(vts,sum);
  creat_double(dts);
  menu(c);
  while (c<>'e')and(c<>'E')do
   begin
   seet:=[];
     case c of
       's','S': begin
                 read_number(n);
                 divi(n);
                end;
       'c','C':begin
                  complet(seet);
```

```
                  print_set(seet);
               end;

       'd','D': begin
                 read_number(num1);
                 read_number(num2);
                 divbetween(num1,num2,seet);
                 print_set(seet);
                end;
       'M','M': begin
                 read_number(num1);
                 read_number(num2);
                 doublebetween(num1,num2,seet);
                 print_set(seet);
                end;
       'p','P':begin
                 pinary(seet);
                 print_set(seet);
               end;
      end;
      menu(c);
   end;
                 print_set(seet);
                end;
       'p','P':begin
                 pinary(seet);
                 print_set(seet);
               end;
      end;
      menu(c);
   end;
End.{End of the program }
```

## المسألة 2:

المطلوب في هذه المسألة التعامل مع الأعداد العقدية. إن بنية المعطيات المستخدمة لتمثيل عدد عقدي هي تسجيلة، لها حقلان: حقل المركبة الحقيقية، وحقل المركبةى التخيلية للعدد العقدي.

البرامج الجزئية التي تحقق العمليات العقدية، والمطلوب برمجتها هي:

Cadd : جمع عددين عقديين.

Csub : طرح عددين عقديين.

Cmul: ضرب عددين عقديين.

Cdiv: قسمة عددين عقديين.

CReead: قراءة عددين عقديين.

CWrite: كتابة عدد عقدي.

CSet: إسناد قيمة لعدد عقدي.

Cabs: تابع يحسب طويلة العدد العقدي.

يقوم البرنامج الأساسي باختيار البرامج الجزئية السابقة: يقوم بقراءة عددين عقديين ويجري العمليات الحسابية، ثم يظهر النتائج بالشكل:

(25.500+50.250 i) + (75.500+10,250 i) = (101.000+60.500 i )

(25.500+50.250 i) - (75.500+10.250 i) = (………………. i )

(25.500+50.250 i) x (75.500+10.250 i) = (………………. i )

(25.500+50.250 i) / (75.500+10.250 i) = (………………. i )

```
TYPE
    COMPLEX=RECORD
       R:REAL;
       C:CHAR;
       I:REAL;
    END;
VAR
  N1,N2,N3:COMPLEX;
  R,F,X,Y:REAL;
  C,C1:CHAR;
FUNCTION  START(C:CHAR):CHAR;
```

```
BEGIN
WRITELN('------------------------------------------------------------------------------------------- |');
WRITELN('|                                                                                             |');
WRITELN('|                     --------------------                                                    |');
WRITELN('|                     |     MAIN MENU    |                                                    |');
WRITELN('|                     --------------------                                                    |');
WRITELN('|         1: TO SUM TWO NUMBER PLEAS PRESS +                                                  |');
WRITELN('|                                                                                             |');
WRITELN('|         2: TO SUB TWO NUMBER PLEAS PRESS -                                                  |');
WRITELN('|                                                                                             |');
WRITELN('|         3: TO MUL TWO NUMBER PLEAS PRESS *                                                  |');
WRITELN('|                                                                                             |');
WRITELN('|         4: TO DIV TWO NUMBER PLEAS PRESS /                                                  |');
WRITELN('|                                                                                             |');
WRITELN('|         5: TO CALULATE LENGTH OF COMPLEX NUMBER |Z| PRESS L                                 |');
WRITELN('|                                                                                             |');
WRITELN('|         6: TO CONVERT ANY COMPLEX NUMBER FORM DECART INTO TRONO FORM PRESS D                |');
WRITELN('|                                                                                             |');
WRITELN('|         7: TO CONVERT ANY COMPLEX NUMBER FORM TRON INTO DECART FORM PRESS  T                |');
WRITELN('|                                                                                             |');
WRITELN('|         8: TO EXIT  PLEAS PRESS E                                                           |');
WRITELN('---------------------------------------------------------------------------------------------');
WRITE(' ENTER THE ONE OF THE FOLOWING CHOICE  ');
READLN(C);
START:=C;
END;

PROCEDURE  CREAD(VAR A:COMPLEX);
VAR
    C:CHAR;
BEGIN
   WRITELN('ENTER THE NUMBER IN THE FOLLOWING WAY: X +IY : ');
   WRITELN('EX:3 +3');
   READ(A.R,C,A.I);
END; {END PROCEDURE}
   WRITELN('ENTER THE NUMBER IN THE FOLLOWING WAY: X +IY : ');
   WRITELN('EX:3 +3');
   READ(A.R,C,A.I);
END; {END PROCEDURE}

FUNCTION CABS(A:COMPLEX):REAL;
VAR
   S:REAL;
BEGIN
   S:=SQRT((A.R*A.R) + (A.I*A.I)); {LENGTH OF COMPLEX NUMBER}
   CABS:=S
END;{END FUNCTION }

PROCEDURE  Cadd(A1,A2:COMPLEX;VAR A3:COMPLEX);
BEGIN
  A3.R:=A1.R+A2.R;
  A3.I:=A1.I+A2.I;
END;{END PROCEDURE}
```

```
PROCEDURE  Csub(A1,A2:COMPLEX;VAR A3:COMPLEX);
BEGIN
  A3.R:=A1.R-A2.R; {VALUE X=R*COS(O)}
  A3.I:=A1.I-A2.I; {VALUE Y=R*SIN(O)}
END;{END PROCEDURE}

PROCEDURE  CONV_TO_TRON(A1:COMPLEX;VAR R,O:REAL);
VAR
   O1:REAL;
BEGIN
  R:= CABS(A1); {R IS LENGTH OF COMPLEX NUMBER}
  O:=ARCTAN((A1.I/A1.R));  {PHASE}
  O:=((O*3.14)/(180)); {CONVERT PHASE TO RADEAN}
END;{END PROCEDURE}

PROCEDURE CONV_TO_DECART(R,O:REAL; VAR A:COMPLEX);
VAR
   O1:REAL;
BEGIN
  O1:=(O*3.14)/(180);
  A.R:=R*COS(O1);
  A.I:=R*SIN(O1);
END;{END PROCEDURE}

PROCEDURE  CMUL(A1,A2:complex;VAR A3:complex);
VAR
  L1,L2,I,I1:INTEGER;
  R1,O1,R2,O2,R3,O3:REAL;
BEGIN
  CONV_TO_TRON(A1,R1,O1);
  CONV_TO_TRON(A2,R2,O2);
  R3:=R1*R2;
  O3:=O1+O2;
  CONV_TO_DECART(R3,O3,A3);
END;{END PROCEDURE}

PROCEDURE CDIV (A1,A2:COMPLEX;VAR A3:COMPLEX);
VAR
   O1,O2,O3,R1,R2,R3:REAL;
BEGIN
  CONV_TO_TRON(A1,R1,O1);
  CONV_TO_TRON(A2,R2,O2);
  R3:=R1/R2;
  O3:=O1-O2;
  CONV_TO_DECART(R3,O3,A3);
END;{END PROCEDURE}
PROCEDURE CWRITE(A1,A2,A3:COMPLEX);
BEGIN
   WRITELN('(',A1.R:2:2,'+',A1.I:2:2,' I',') ', C , '(',A2.R:2:2,'+',A2.I:2:2,'
I',')','=','(',A3.R:2:2,'+',A3.I:2:2,' I',')');
END;{END PROCEDURE}
```

```
BEGIN {**************************MAIN PROGRAM******************************}
REPEAT
C:=START(C); {MAIN MENU}
IF C='+' THEN
BEGIN
  CREAD(N1);
  CREAD(N2);     {SUM TWO NUMBER }
  CADD(N1,N2,N3);
  CWRITE(N1,N2,N3);
END;

IF C='-' THEN
BEGIN
  CREAD(N1);
  CREAD(N2);   { SUB TWO NUMBER }
  CSUB(N1,N2,N3);
  CWRITE(N1,N2,N3);
END;

IF C='*' THEN
BEGIN
  CREAD(N1);
  CREAD(N2);   { MUL TWO NUMBER }
  CMUL(N1,N2,N3);
  CWRITE(N1,N2,N3);
End;

IF C='/' THEN
BEGIN
  CREAD(N1);
  CREAD(N2);   { DIV TWO NUMBER }
  CDIV(N1,N2,N3);
  CWRITE(N1,N2,N3);
END;{END IF}

IF C IN['L','l'] THEN
BEGIN
    CREAD(N1); {TO FIND LENGTH OF COMPLEX NUMBER}
    WRITELN('LENGTH OF  |',N1.R:2:2,' + ',N1.I:2:2,' I','|','  IS =',CABS(N1):2:2);
END;{END IF}

IF  C IN['D','d'] THEN
BEGIN  {CONVERT ANY COMPEX NUMBER FROM DECART FORM INTO TRON FORM}
  WRITE('ENTER THE X=');READLN(N1.R);
  WRITE('ENTER THE Y=');READLN(N1.I);
  CONV_TO_TRON(N1,R,F);
  WRITELN('[',R:2:2,' , ',F:2:2,']');
END;{EN IF}

IF  C IN['T','t'] THEN
BEGIN {CONVERT ANY COMPLEX NUMBER FROM TRON FORM INTO DECART FORM}
```

```
  WRITE('ENTER THE R='); READLN(R);
  WRITE('ENTER THE PHASE ='); READLN(F);
  CONV_TO_DECART(R,F,N1);
  WRITELN('(',N1.R:2:2,'+',N1.I:2:2,' I ',')');
END;{EN IF}
IF  C IN['E','e'] THEN
 BEGIN
   WRITE(' ARE YOU SURE TO EXIT ,TYPE Y/N ?');

   READLN(C1);
  END;
UNTIL  ((C IN['E','e']) AND (C1 IN['Y','y']));
END.
   READLN(C1);
  END;
UNTIL  ((C IN['E','e']) AND (C1 IN['Y','y']));
END.
```

## المسألة 3:

المطلوب في هذه المسألة التعامل مع التاريخ. بنية المعطيات المستخدمة لتمثيل التاريخ هي تسجيلة بالشكل

```
Type date= Record
    Year,day,mounth: integer;
end ;
```

المطلوب كتابة البرامج الجزئية التالية :

Set Date: بالنمط تاريخ ثلاث قيم صحيحة تمثل اليوم والشهر والسنة والعام.

WriteDate: يظهر التاريخ على الشاشة بالشكل المألوف، أي 31/10/1999

GreaterDate : تابع منطقي يقارن بين تاريخين ويعطي القيمة صح إن كان الأول أكبر تماماً من الثاني.

BetweenDate: يعطي الفرق بين تاريخين.

يقوم البرنامج الأساسي باستدعاء البرامج الجزئية السابقة واختيارها.

```
  TYPE
  DATE=RECORD
    DAY:INTEGER;
    MONTH:INTEGER;
    YEAR:INTEGER;
  END;

VAR
DAT1,DAT2:DATE;
F:BOOLEAN;
PROCEDURE   READDATE(VAR D1:DATE);
VAR S,SLASH:CHAR;

BEGIN

WRITE('ENTER THE DATE ON THE FOLLOWING WAY : DAY /MONTH /YEAR:');
READLN(D1.DAY,S,SLASH,D1.MONTH,S,SLASH,D1.YEAR);
END;

PROCEDURE WRITEDATE(D1:DATE);
BEGIN
         WRITELN('THE DATE IS :',D1.DAY,'/',D1.MONTH,'/',D1.YEAR);
END;

FUNCTION GREATERDATE(D1,D2:DATE):BOOLEAN;
         VAR  RESULT:BOOLEAN;
BEGIN
  RESULT:=FALSE;
  IF D1.YEAR>D2.YEAR THEN
```

```
     RESULT:=TRUE
  ELSE IF D1.YEAR=D2.YEAR THEN
      IF D1.MONTH>D2.MONTH THEN
           RESULT:=TRUE
      ELSE  IF D1.MONTH=D2.MONTH THEN
            IF D1.DAY>D2.DAY THEN
              RESULT:=TRUE;

GREATERDATE:=RESULT;
END;

PROCEDURE BETWEENDATE(D1,D2:DATE);
         VAR  D3:DATE;
BEGIN
         D3.YEAR:=D1.YEAR-D2.YEAR;
         D3.MONTH:=D1.MONTH-D2.MONTH;
         D3.DAY:=D1.DAY-D2.DAY;
{WRITELN('THE DIFFERENCE  BETWEEN  TWO DATE IS : ',D3.DAY,'/',D3.MONTH,'/',D3.YEAR);}
WRITELN('THE DIFFERENCE  BETWEEN  TWO DATE IS : ',D3.YEAR,' YEARS','  AND ',D3.MONTH,' MONTH',' AND
',D3.DAY,' DAY');
END;

  ELSE IF D1.YEAR=D2.YEAR THEN
      IF D1.MONTH>D2.MONTH THEN
           RESULT:=TRUE
      ELSE  IF D1.MONTH=D2.MONTH THEN
            IF D1.DAY>D2.DAY THEN
              RESULT:=TRUE;
GREATERDATE:=RESULT;
END;
PROCEDURE BETWEENDATE(D1,D2:DATE);
         VAR  D3:DATE;
BEGIN
         D3.YEAR:=D1.YEAR-D2.YEAR;
         D3.MONTH:=D1.MONTH-D2.MONTH;
         D3.DAY:=D1.DAY-D2.DAY;
{WRITELN('THE DIFFERENCE  BETWEEN  TWO DATE IS :',D3.DAY,'/',D3.MONTH,'/',D3.YEAR);}
WRITELN('THE DIFFERENCE  BETWEEN  TWO DATE IS : ',D3.YEAR,' YEARS','  AND ',D3.MONTH,' MONTH',' AND
',D3.DAY,' DAY');
END;
```

## المسألة 4:

المطلوب في هذه المسألة حساب علامات الطلب وإصدار قوائمها.

يُعرّف كل طالب برقم ذاتي (عدد صحيح موجب)، واسم(سلسلة حرفية لا تتجاوز 25حرفاً )، إضافة إلى علامات المواد: رياضيات ، برمجة، لغة عربية، لغة أجنبية. تُحسب العلامة من100.

المطلوب قراءة المعلومات الخاصة بالطلاب وعلاماتهم، وحساب معدلاتهم وإصدار قوائم العلامات ملاتبة وفق المعدلات. كما يطلب حساب وسطي العلامات والانحراف المعياري في كل مادة.

```
  type
    studentmark=record
      math:integer;
      programing:integer;
      arabic:integer;
    end;

    student=record
      serialnumber:integer;
      name:string[15];
      stdmark:studentmark;
      average:real;
      leavel:string;
    end;

var
    std_mark:array[1..100] of  student;
    n,i,j:integer;    temp:student;
    s_math,s_programing,s_arabic,s_all,s_lest : set of 1..100;

begin
var
    std_mark:array[1..100] of  student;
    n,i,j:integer;    temp:student;
    s_math,s_programing,s_arabic,s_all,s_lest : set of 1..100;
begin
                  write(' enter the number of student:='); readln(n);
                  for i:= 1 to n do
                   begin
                     writeln('enter the data of student  ',i);
                     writeln('------------------------------');
                      with std_mark[i]  do
                        begin
                           write('enter the serial number=');
                            readln(serialnumber);
                           write(' enter the name of the student: ');
                            readln(name);
```

```
                        write(' enter the grade of math=');
                         readln(stdmark.math);
                        write(' enter the grade of programing=');
                         readln(stdmark.programing);
                        write(' enter the grade of arabic='); readln(stdmark.arabic);
                      end;
                 writeln('-------------------------------');
                end;
               s_math:=[];
               s_programing:=[];
               s_arabic:=[];
               s_all:=[];
               s_lest:=[];


               for i:= 1 to n do
                 begin
                   if std_mark[i].stdmark.math>=60 then
                      s_math:=s_math+[std_mark[i].serialnumber];
                   if std_mark[i].stdmark.programing>=60 then
                      s_programing:= s_programing+[std_mark[i].serialnumber];
                   if std_mark[i].stdmark.arabic>=60 then
                      s_arabic:=s_arabic+[std_mark[i].serialnumber];
                 end;

               s_lest:=s_math+s_programing+s_arabic;
               s_all:=s_math*s_programing*s_arabic;
               j:=1;
               writeln('sn':10,'name':10,'math':10,'programing':20,'arabic':10);
                 end;
               s_lest:=s_math+s_programing+s_arabic;
               s_all:=s_math*s_programing*s_arabic;
               j:=1;
               writeln('sn':10,'name':10,'math':10,'programing':20,'arabic':10);
               while ((j<=n) and (s_lest<>[])) do
               begin
                 if ( std_mark[j].serialnumber in s_lest) then
                    begin
                      with std_mark[j] do
                         begin
writeln(serialnumber:10,name:10,stdmark.math:10,stdmark.programing:20,stdmark.arabic:10);
                          end;
                      s_lest:=s_lest-[std_mark[j].serialnumber];
                    end;
                        j:=j+1;
               end;


               for i:= 1 to n do
                begin
                   std_mark[i].average:=0;
                 if std_mark[i].serialnumber in s_all then
                    begin
```

```
                           with std_mark[i] do
std_mark[i].average:=((stdmark.math+stdmark.programing+stdmark.arabic) /3) ;
                             case round(std_mark[i].average) of
                                             60..69: std_mark[i].leavel:='good';
                                             70..79: std_mark[i].leavel:='very good';
                                             80..100:std_mark[i].leavel:='exclent';
                                  end;
                           end
                       else
                         std_mark[i].average:=0;    end; { End For LOOP}
                   end;
                   for i:= 1 to n-1 do
                   begin
                     for j:=i+1 to n do
                       begin
                          if std_mark[i].average>std_mark[j].average then
                            begin
                               temp:=std_mark[i];
                               std_mark[i]:=std_mark[j];
                               std_mark[j]:=temp;
                            end;
                       end;
                   end;
                   writeln('-------------------',' students sorting ' ,'-------------------------');

         writeln('sn':10,'name':10,'math':10,'programing':20,'arabic':10,'average':10,'leavel':10);


                 for i:= 1 to n do
                   begin
                     with std_mark[i] do
write(serialnumber:10,name:10,stdmark.math:10,stdmark.programing:17,stdmark.arabic:10,average:12:2,'
':4,leavel);
                     writeln;
                   end;
                   readln;
end.
```

## المسألة 5 و 6:

المطلوب كتابة الإجرائيات التالية:

1. إجرائية تحسب عدد الكلمات في كل سطر.
2. إجرائية عدد مرات تكرار كلمة في نص ضمن الملف.
3. إجرائية إستبدال كلمة word1(موجودة في النص) بكلمة word2.

إجرائية عدد الكلمات في كل سطر

```
procedure num_words(var A:text; var nw:integer);
Var
        c:char; l:integer;
Begin
        Assign(A,'c:\textfile');
        reset(A);
        While not eof(A) do
        Begin
                read(A,c);
                nw:=0;
                l:=l+1;
                While not eoln(A) do
                Begin
                        While (ord(c)=32) do
                        Begin
                                read(A,c);
                                if (ord(c) in [65..90,97..122]) then
                                        nw:=nw+1;
                        End;
                        read(A,c);
                End;
            writeln('the number of word in the line ',l ,' is =', nw);
        End;
End;
```

إجرائية عدد مرات تكرار كلمة في نص ضمن الملف:

```
procedure word_freq(Var A:text; word:string; var r:integer);
Var
        c:char; s:string;
Begin
        Assign(A,'c:\textFile');
        reset(A);
        While not eof(A) do
        Begin
                read(A,c);
                s:='';
                while not eoln(A) do
                Begin
                        while (ord(c)<>32) do
                        Begin
                                s:=s+c;
                                read(A,c);
                        End;
                        if s=word then
                                r:=r+1;
```

```
                        s:='';
                        Read(A,c);
                End;
        End;

End;
```

إجرائية إستبدال كلمة word1(موجودة في النص) بكلمة word2

```
procedure replace_word(Var A,B:text; word1,word2:string);
Var
        c:char; s:string;
Begin
        Assign(A ,'c:\A ');
        reset(A);
        Assign(B,'c:\b');
        rewrite(B);
        While not eof(A) do
        Begin
                read(A,c);
                s:='';
                while not eoln(A) do
                Begin
                        while (ord(c)<>32) do
                        Begin
                                write(B,c);
                                s:=s+c;
                                read(A,c);
                        End;
                        if s=word1 then
                                write(B,word2);
                    s:='';
                        Read(A,c);
                End;
        End;
End;
```

# المسألة 7:

المطلوب قراءة ملف نصي وكتابته في ملف نصي آخر بعد تحويل كل حرف صغير فيه إلى حرف كبير.

```
 VAR
        T1,T2:TEXT;
        C,C1:CHAR;
        ORDE:INTEGER;
BEGIN
        ASSIGN(T1,'C:\S_LETTER.TXT');
        ASSIGN(T2,'C:\C_LETTER.TXT');
        REWRITE(T1);
        WRITELN(' ENTER THE TEXT WHICH END WITH . ');
        WHILE C<>'.' DO
        BEGIN
          READ(C);
          WRITE(T1,C);
        END;
        CLOSE(T1);
        RESET(T1);
        REWRITE(T2);
        WHILE NOT(EOF(T1)) DO
        BEGIN
                 READ(T1,C);
                 ORDE:=ORD(C);
           CASE ORDE OF
                97..122 : BEGIN     C1:=CHR(ORDE-32);    WRITE(T2,c1);     END;
                65..90 : WRITE(T2,C);
            ELSE
                WRITE(T2,C);
            END;
        END;
        CLOSE(T1);
        CLOSE(T2);
        RESET(T2);
        WRITELN('*******************************************************');
        WRITELN('*****************  THE  TWO TEXT IS********************');
        WRITELN('*******************************************************');
        CLOSE(T2);
        RESET(T2);
        WRITELN('*******************************************************');
        WRITELN('*****************  THE  TWO TEXT IS********************');
        WRITELN('*******************************************************');
        WHILE NOT(EOF(T2)) DO
        BEGIN
                 READ(T2,C);
                 WRITE(C);
        END;
        CLOSE(T2);
        READLN;
        READLN;
END.
```

# المسألة 8:

اكتب برنامجاً يقوم بالعمليات التالية:

أ- الخزن في ملف ثنائي لمعطيات تعريف الكتب في مكتبة. يوصف الكتاب برقم تسلسلي، وعنوان واسم المؤلف، وتاريخ الإصدار، لإضافة إلى حقل منطقي يدل على الإعارة، يقوم البرنامج بقراءة المعلومات من ملف معطيات نصي.

ب- تسجيل الإعارة أو الإعادة في الملف. يقوم المستخدم بإعطاء جدول بأرقام الكتب المعارة والمعادة ليسجلها البرنامج على الملف.

ت- إظهار لائحة بالكتب المعارة.

```
  TYPE DATE=RECORD
       DAY:INTEGER;
       MONTH:INTEGER;
       YEAR:INTEGER;
     END;
   LIB=RECORD
     SN:INTEGER;
     TITEL:STRING[7];
     NAME:STRING[8];
     DAT:DATE;
     BORROW:BOOLEAN;
   END;
   BOR=RECORD
     BORROW:INTEGER;
     NOTBORROW:INTEGER;
   END;

VAR
         BF,B1:FILE OF LIB;  T1:TEXT;
         D1:DATE;  TEMP,R:LIB;
         C:CHAR;
         A:ARRAY[1..100] OF BOR; I,N:INTEGER;
         S_N:SET OF 1..100;    B:BOOLEAN;
         S1,S2:STRING[7];
BEGIN
                  ASSIGN(BF,'k:\LIBRARAY');
                  ASSIGN(T1,'k:\T1.TXT');
                  ASSIGN(B1,'k:\TEMP');
                  RESET(T1);
                  REWRITE(BF);                 {1}
                  WHILE NOT(EOF(T1)) DO
                   BEGIN
```

```
                    READLN(T1,R.SN,R.TITEL,R.NAME,R.DAT.DAY,R.DAT.MONTH,R.DAT.YEAR);
                    WRITE(BF,R);
                   END;
                  CLOSE(T1);
                  CLOSE(BF);
                  {-----------------------------------------------------------------------}
                  RESET(BF);
                  REWRITE(B1);
                  WHILE NOT(EOF(BF)) DO
                  BEGIN
                  READ(BF,R);            {SWAP BETWEEN TWO FILES}
                  WRITE(B1,R);
                  END;
                  CLOSE(BF);
                  CLOSE(B1);
                  {------------------------------------------------------------------------}
                  CLOSE(BF);
                  CLOSE(B1);
                  {------------------------------------------------------------------------}
                  S_N:=[];
                  REWRITE(BF);
                  RESET(B1);
                  WRITE('ENTER THE NUMBER OF BOOKS '); READLN(N);
         WRITELN('ENTER THE NUMBERS OF THE BOOKS  WITCH BORROW ');
                  FOR I:=1 TO N DO                  {2}
                  BEGIN
                    WRITE('THE BOOK ',I,'=');
                    READLN(A[I].BORROW);          {READING LIST 1}
                    WHILE (NOT(EOF(B1))AND (A[I].BORROW<>TEMP.SN))  DO
                      BEGIN
                          READ(B1,TEMP);
                          IF TEMP.SN=A[I].BORROW THEN
                          BEGIN
                               TEMP.BORROW:=TRUE;        {3}
                               WRITE(BF,TEMP);
                               S_N:=S_N+[TEMP.SN]
                          END;
                      END;{END WHILE}
                  END;{END FOR}
                  {-----------------------------------------------------------------------}
WRITELN('ENTER THE NUMBERS OF THE BOOKS  WITCH NOTBORROW ');
                  FOR I:=1 TO N DO
                  BEGIN
                     WRITE('THE BOOK ',I,'=');        {READING LIST 2}
                     READLN(A[I].NOTBORROW);
                  END;{END FOR}
                  CLOSE(B1);
                  CLOSE(BF);
                  {-----------------------------------------------------------------------}
                  writeln('-----------------------------------------------------------------');
WRITE('ENTER THE ADDRESS OF THE BOOK WHICH YOUSEARCH IT :');
READLN(S1);
                  RESET(B1);
```

```
                   REPEAT
                   S1:=' '+S1;
                   UNTIL LENGTH (S1)=7;
                   IF R.TITEL=S1 THEN
                     IF R.SN IN S_N THEN
                        WRITELN('THE BOOK  ',S1,' CAN NOT BE BORROWED')
                     ELSE
                         WRITELN('THE BOOK  ',S1,' CAN  BE BORROWED');
                   WHILE ((NOT(EOF(B1))) AND (S1<>R.TITEL)) DO
                   BEGIN
                     READ(B1,R);                    {4}
                     IF R.TITEL=S1 THEN
                         IF R.SN IN S_N THEN
                             WRITELN('THE BOOK  ',S1,' CAN NOT BE BORROWED')
                         ELSE
                             WRITELN('THE BOOK  ',S1,' CAN  BE BORROWED')
                   END;
                   CLOSE(B1);
                   {-----------------------------------------------------------------------}
                   writeln('------------------------------------------------------------');
                   WRITELN('ENTER THE DATE ');
                   WRITELN('----------------');
                   WRITE('ENTER THE DAY='); READLN(D1.DAY);
                   WRITE('ENTER THE MONTH='); READLN(D1.MONTH);
                   WRITE('ENTER THE YEAR='); READLN(D1.YEAR);
                   RESET(B1);
WRITELN(' THE BOOK WHITCH ARE CREATER BEFOR DATE ',D1.DAY,'\',D1.MONTH,'\',D1.YEAR,' :');
WRITELN('------------------------------------------------------------------------------');
                   WHILE NOT(EOF(B1)) DO
                   BEGIN
                     READ(B1,R);
                      B:=FALSE;                              {5}
                      IF D1.YEAR>R.DAT.YEAR THEN
                         B:=TRUE
                      ELSE IF D1.YEAR=R.DAT.YEAR THEN
                         IF D1.MONTH>R.DAT.MONTH THEN
                                 B:=TRUE
                          ELSE IF D1.MONTH=R.DAT.MONTH THEN
                                IF D1.DAY>R.DAT.DAY THEN
                                  B:=TRUE;
                      IF B=TRUE THEN

WRITELN(R.SN,C,R.TITEL,C,R.NAME,C,R.DAT.DAY:2,R.DAT.MONTH:4,R.DAT.YEAR:8,R.BORROW:8);
                   END;
                   BEGIN{*}
                     READ(B1,R);
                      B:=FALSE;                              {5}
                      IF D1.YEAR>R.DAT.YEAR THEN
                         B:=TRUE
                      ELSE IF D1.YEAR=R.DAT.YEAR THEN
                         IF D1.MONTH>R.DAT.MONTH THEN
                                 B:=TRUE
                          ELSE IF D1.MONTH=R.DAT.MONTH THEN
```

```
                                 IF D1.DAY>R.DAT.DAY THEN
                                   B:=TRUE;
                       IF B=TRUE  THEN
 WRITELN(R.SN,C,R.TITEL,C,R.NAME,C,R.DAT.DAY:2,R.DAT.MONTH:4,R.DAT.YEAR:8,R.BORROW:8);
                  END; {End *}
                  RESET(BF);
WRITELN(' THE BOOKS WHICH ARE  BORROWED IN THE LIBRARAY IS....');
                  WHILE NOT(EOF(BF)) DO
                  BEGIN                        {PRINT LIST}
                  READ(BF,R);
         WRITELN(R.SN,C,R.TITEL,C,R.NAME,C,R.DAT.DAY:2,R.DAT.MONTH:4,R.DAT.YEAR:8,R.BORROW:8);
                  END;
                  CLOSE(BF);
                  WRITELN('--------------------------------------------------------------');
                  RESET(B1);
                  WHILE NOT(EOF(B1)) DO
                  BEGIN                        {PRINT LIST}
                  READ(B1,R);
         WRITELN(R.SN,C,R.TITEL,C,R.NAME,C,R.DAT.DAY:2,R.DAT.MONTH:4,R.DAT.YEAR:8,R.BORROW:8);
                  END;
                  CLOSE(B1);
                  READLN;
END.
```

## المسألة 9:

لدينا ملفان ثنائيان من الأعداد الصحيحة مرتبان تصاعدياً، نريد دمجهما بحيث نحصل على ملف واحد مرتب يحوي عناصر الملفين. سنقوم بالدمج دون استخدام الجداول في البرنامج لأننا نفترض أن الملفان كبيرين جداً، و هذا مانسميه بالفرز الخارجي.

اكتب أولاً برنامجاً ينشئ الملفات الثنائية من ملفات نصية تحوي أعداداً صحيحة.

ثم اكتب برنامج الدمج الذي ينشئ الملف الثنائي الناتج، ثم نقوم بكتابته أيضاً في ملف نصي.

```
Procedure merge2file(var in1,in2:text; var  out:text);
Var
         n1,n2:integer;
         Function getvalue (var f:text):integer;
                  Var  n :integer;
         Begin
                  if not eof(in1)then
                  Read(f,n)
                  Else
                  N:=maxint;
                  Getvalue:=n;
         End;
Begin
         Reset(in1);
         Reset(in2);
         Rewrite(out);
         N1:=getvalue(in1);
         N2:=getvalue(in2);
         While (n1< maxint) or (n2 <maxint) do
                  If n1<n2 then
                  Begin
                           Write(out,n1);
                           N1:=getvalue(in1);
                  End
                  Else
                  Begin
                           Write(out,n2);
                           N2:=getvalue(in2);
                  End;
         Close(in1);
         Close(in2);
         Close(out);
End;
```

## المسألة 10:

المطلوب بناء برنامج لتحقيق لائحة معطيات مزدوجة الترابط(double linked list)، باستخدام بنية تحوي مؤشرين بدلاً من مؤشر واحد كما في اللائحة أحادية الترابط ، مؤشر إلى العنصر السابق ومؤشر إلى العنصر اللاحق. كما  نستخدم مؤشرين الأول يشير إلأى الرأس والثاني إلى الذيل.

```
Type
         Dl_p=^Dl_r;
         Dl_r=record
                  key:integer;
                  next:Dl_p;
                  last:Dl_p;
         End;
procedure insert(var ls,le:Dl_p; key:integer);
var
         temp,s:Dl_p;
         located:boolean;
begin
         new(temp);
         temp^.key:=key;
         temp^.next:=nil;
         temp^.last:=nil;
         if ls=nil then
         Begin
                  ls:=temp;
                  le:=temp;
         End
         Else
         Begin
         Begin
                  s:=ls;
                  located:=false;
                  while ((s<>nil) and (not(located))) do
                           if s^.key<key then
                                    s:=s^.next
                           Else
                                    located:=true;
                  temp^.next:=s;
                  if s=ls then {}
                  Begin
                           temp^.next:=ls;
                           ls:=temp;
                  End
                  Else
                  Begin
                           if s=nil then
                           Begin
                                    le^.next:=temp;
                                    temp^.last:=le;
                                    le:=temp;
                           End
```

```
                                Else
                                Begin
                                        temp^.last:=s^.last;
                                        s^.last:=temp;
                                        temp^.last^.next:=temp;
                                End;
                        End;
                End;
End;
procedure Delete(var ls,le:Dl_p; key:integer);
var temp,s:Dl_p;
Begin
        if ls=nil then
                writeln('the Double linked list is Empty')
        Else
                if ls^.key=key then
                Begin
                        temp:=ls;
                        ls:=ls^.next;
                        Dispose(temp);
                End
                Else
                Begin
                        if le^.key=key then
                        Begin
                                temp:=le;
                                le:=le^.last;
                                le^.next:=nil;
                                Dispose(temp);
                        End
                        Else
                        Begin
                                s:=ls;
                                while ((s<>nil) and (s^.key<>key)) do
                                        s:=s^.next;
                                if s=nil then
                                        writeln('the Element is not found in the list')
                                Else
                                Begin
                                        s^.last^.next:=s^.next;
                                        s^.next^.last:=s^.last;
                                        Dispose(temp);
                                End;
                        End;
                End;

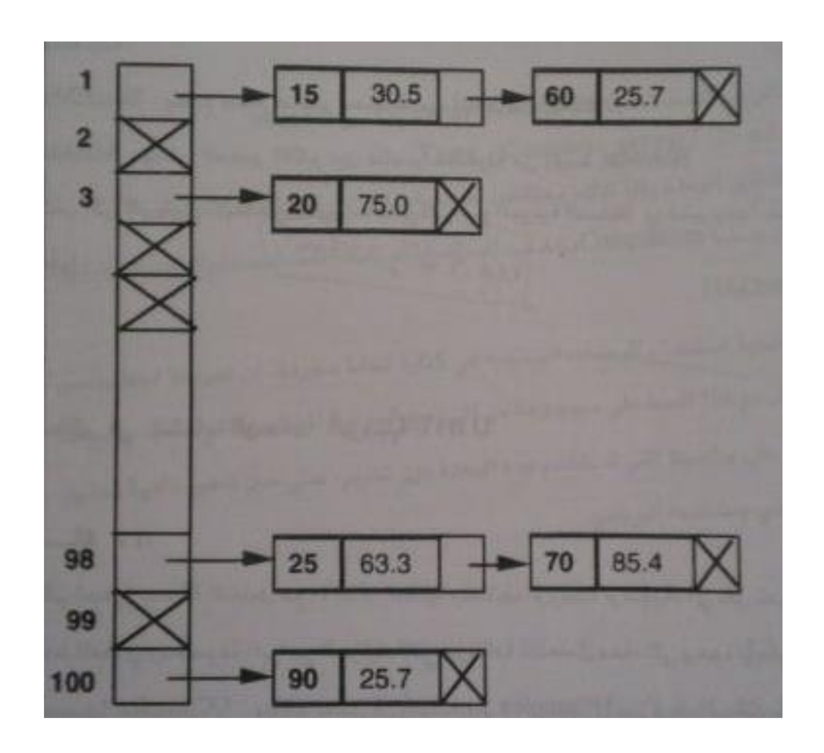
End;
                                        End;
                                End;
                        End;
End;
```

## المسألة 11:

تتعامل مسألتنا مع ما يسمى "المصفوفات المثقوبة"، وهي مصفوفات تغلب فيها القيمة صفر، فمحدودة هي القيم المختلفة عن الصفر.ويهدف إلى الاختصار في الذاكرة اللازمة لمصفوفات كبيرة من هذا النوع، سنستخدم المؤشرات المرسومة في الشكل المرفق، والمعرّفة كما يلي:

```
Const MaxN= 100;
Type PEMAT= ^EMAT;
Type EMAT= record
Begin
Col: integer;
Var: real;
Penxt: PEMAT;
End
Type HoleMat= array [1..MaxN] of PEMAT;
```

وبهذا تكون المصفوفة جدولاً من المؤشرات يؤشر كل منها إلى سلسلة العناصر غير الصفرية في سطر المصفوفة، ويكون عنصر المصفوفة ممثلاً بتسجيلة، عناصرها: رقم العمود وقيمة العنصر ومؤشر إلى عنصر آخرفي السطر نفسه.الصورة التالية تبين تمثيل مصفوفة مثقوبة باستخدام المؤشرات.




والآن وقد عرفنا طريقة تعريف نمط المصفوفة يطلب كتابة البرامج الجزئية التالية:

ReadHMat: يقرؤأ عناصر المصفوفة من الدخل ويخزنها في متحول من نمط HoleMat.

WriteHMat: يكتب مصفوفة من النمط HoleMat (العناصر المختلفة عن الصفر فقط).

AddHMat: يجمع مصفوفتين من النمط HoleMat. ويضع الناتج في مصفوفة من النمط ذاته.

SumHMat: يجمع جميع عناصر مصفوفة من النمط HoleMat.

MaxMat: يعطي العنصر الأكبر بين عناصر مصفوفة من النمط HoleMat.

اكتب الآن البرنامج الأساسي الذي يستدعي البرامج الجزئية السابقة ويختبر جيداً صحة عملها.

```
  Const MaxN=100;
Type Pemat=^Emat;
     Emat=Record
       Col:integer;
       Val:Real;
       Pnext:Pemat;
     End;
    Holmat=Array[1..maxn] of pemat;
Var
   ls:pemat;
   n,m,i:integer;
   h_mat,h_mat1,h_mat2,h_mat3,h_mat4,h_mat5:holmat;
   C:Char;
   S,g:Real;
{---------------------------------------------------------------------}
PROCEDURE SPACE;
VAR
   I:INTEGER;
BEGIN
   FOR I:= 1 TO 100 DO
      WRITELN;
END;{END PROCEDURE}
{----------------------------------------------------------------------}
PROCEDURE MENU(VAR C:CHAR);
BEGIN
  SPACE;
  Writeln('|------------------------------------------------------------- -|');
  Writeln('|                                                               |');
  Writeln('|             WELCOME IN MATRIX  HOLED PROGRAM                  |');
  Writeln('|                                                               |');
  Writeln('|                                                               |');
  Writeln('|                   WHAT DO YOU WANT TO DO ?                    |');
  Writeln('|                                                               |');
  Writeln('|                    -------------------                        |');
  Writeln('|                    |    MAIN MENU    |                        |');
  Writeln('|                    -------------------                        |');
  Writeln('|                                                               |');
  Writeln('|                                                               |');
  Writeln('|                1: To Read Array Elements.                     |');
  Writeln('|                                                               |');
  Writeln('|                                                               |');
  Writeln('|                2: To Print Array Elements.                    |');
```

```
  Writeln('|                                                       |');
  Writeln('|                                                       |');
  Writeln('|             3: To Sum Two Array.                      |');
  Writeln('|                                                      |');
  Writeln('|                                                      |');
  Writeln('|             4: To Sum Elements of Array.             |');
  Writeln('|                                                      |');
  Writeln('|                                                      |');
  Writeln('|             5: To Get Max Elements in Array .        |');
  Writeln('|                                                      |');
  Writeln('|                                                      |');
  Writeln('|             6: TO EXIT FROM PROGRAM.                 |');
  Writeln('|                                                      |');
  Writeln('|-----------------------------------------------------|');
  Writeln(' ENTER ONE OF THE FOLOWING CHOICES ');
  Readln(C);
End;{End Procedure}
{----------------------------------------------------------------------------}
Procedure  Insert(Var ls:pemat; R:emat);
Var t,p,prev:pemat;
Begin
   New(t);
   t^:=R;
   t^.Pnext:=Nil;
   If Ls=Nil Then
      Ls:=t
   Else {*}
   Begin
     If t^.Col<ls^.Col Then
       Begin
         t^.Pnext:=ls;
         ls:=t;
        End
     Else {**}
     Begin
        p:=Ls;
        while ((p<>Nil) And (t^.Col>p^.Col)) Do
         Begin
           prev:=p;
           p:=p^.Pnext;
         End;
         If  (t^.Col<>p^.Col) Then
           Begin
             t^.Pnext:=p;
             prev^.Pnext:=t;
           End
           Else
              p^.val:=p^.val+t^.val;
     End;{End **}
   End;{End *}
End;{End Procedure}
{---------------------------------------------------------------------------}
Procedure Readline_h_mat(Var ls:pemat; n:integer);
```

```
Var R:emat;
Begin
   Writeln(' To Start Press Enter , But To Exit press CTRL+Z ... ');
   While Not Eof Do
   Begin                            {1}
      Readln;
      Repeat
         Write('Enter The Column number ='); Readln(R.Col);
      Until R.col<=n;
      Repeat
         Write('Enter The value='); Readln(R.Val);
      Until R.val<>0;
      Insert(ls,R);
      Writeln(' To Continue Press Enter , But To Exit press CTRL+Z ... ');
  End; {End While}
End;{End Procedure}
{--------------------------------------------------------------------------}
Procedure Read_holmat(Var hol:Holmat; m,n:integer);
Var i:integer;
Begin
 For i:= 1 To m do
  Begin
    Writeln('Enter the element of line ',i);
    Readline_h_mat(hol[i],n);
  End;{End For}
End;{End Procedure}
{--------------------------------------------------------------------------}
Procedure Add_hmat(hol1,hol2:holmat;  Var hol3:holmat; m,n:integer );
Var i:integer; R:Emat; p1,p2:Pemat;
Begin
    For i:= 1 to m do
       hol3[i]:=nil;
    For i:= 1 to m do
     Begin {Merge to list ; if (p1^.col=p2^.col) then sum between then}
       hol3[i]:=nil;
    For i:= 1 to m do
     Begin {Merge to list ; if (p1^.col=p2^.col) then sum between then}
             p1:=hol1[i];  p2:=hol2[i];
             While ((p1<>nil) or (p2<>nil)) do
              Begin
                 if p1^.col=p2^.col then
                 Begin
                  R.Val:=((p1^.Val) + (p2^.Val));
                  R.Col:=p1^.Col;
                  insert(hol3[i],R);
                  p1:=p1^.pnext;
                  p2:=p2^.pnext;
                 End
                Else
                    if p1^.Col<p2^.Col then
                    Begin
                     R:=p1^;
                     Insert(hol3[i],R);
```

```
                     p1:=p1^.pnext;
                    End
                  Else
                    if p2^.Col<p1^.Col then
                    Begin
                     R:=p2^;
                     Insert(hol3[i],R);
                     p2:=p2^.pnext;
                    End;
              End;{End While}
    End;{End For}
End;{End Procedure}
Procedure sum_Elements(hol:holmat; m:integer; Var Sum:Real );
Var i:integer; p:pemat;
Begin
  Sum:=0;
  For i:= 1 to m do
   Begin
      p:=hol[i];
      While p<>nil do
      Begin
        sum:=sum+p^.val;
        p:=p^.pnext;
      End;{End While}
   End;{End For}
End;{End Procedure}
{-----------------------------------------------------------------------------}
Procedure Greatest(hol:holmat; m:integer;  var great:Real);
Var  i:integer;  p:pemat;
Begin
   great:=0;
  For i:= 1 to n do
   Begin
      p:=hol[i];
      While p<>nil do
      Begin
        if p^.val>great then
            great:=p^.val;
        p:=p^.pnext;
      End;{End While}
   End;{End For}
End;{End Procedure}
Procedure  Display(Var hol:holmat; m,n:integer);
Var i,j:integer;
    p:pemat;
Begin
    For i:= 1 To m  do
    Begin
        p:=hol[i];
        For j:= 1 To n do
          If (((p<>nil) or (p=nil)) and(j<>p^.col)) then
             Write(0:6)
          Else
```

```
            Begin
              Write(p^.val:6:2);
              p:=p^.pnext;
            End;{End Else}
        Writeln;
        End;{End For}
End;{End Procedure}
Begin{****************************Main Program******************************}
Repeat
  Menu(c);
  Case  C of
    '1':
     Begin
       Write('Enter the line number=');   Readln(m);
       Write('Enter the column number='); Readln(n);
       For i:= 1 To m Do
           h_mat[i]:=nil;
       Read_holmat(h_mat,m,n);
       Write('Press Enter To main Menu.......');
       Readln;
     End;
     '2':
      Begin
        Display(h_mat,m,n);
        Write('Press Enter To main Menu.......');
        Readln;
      End;
      '3':
       Begin { First Array}
          Writeln(' Enter the  First Array......');
          Writeln('--------------------------------------------');
          Write('Enter the line number For Two Array [A] & [B] =');   Readln(m);
          Write('Enter the column number For Two Array [A] & [B] ='); Readln(n);
          For i:= 1 To m Do
             h_mat1[i]:=nil;
          Read_holmat(h_mat1,m,n);
          { Second Array }
          Writeln('Enter the Second Array.......');
          Writeln('--------------------------------------------');
         { Write('Enter the line number=');   Readln(m);
          Write('Enter the column number='); Readln(n);}
          For i:= 1 To m Do
              h_mat2[i]:=nil;
          Read_holmat(h_mat2,m,n);
          { Result Array [A]+[B]}
          {Notice!!!!:[A] and [B] Must Be Both of Them is Same type and form}
          Add_hmat(h_mat1,h_mat2,h_mat3,m,n); {Sum Procedure}
          Writeln('************The First Array [A]************');
          Writeln('--------------------------------------------');
          Display(h_mat1,m,n);
          Writeln('************The Second Array [B]************');
          Writeln('--------------------------------------------');
          Display(h_mat2,m,n);
```

```
          Writeln('************The Result Array [A]+[B]************');
          Writeln('--------------------------------------------');
          Display(h_mat3,m,n);
          Write('Press Enter To main Menu.......');
          Readln;
       End;
          Writeln('--------------------------------------------');
          Display(h_mat3,m,n);
          Write('Press Enter To main Menu.......');
          Readln;
       End;
      '4':
       Begin
          Write('Enter the line number=');   Readln(m);
          Write('Enter the column number='); Readln(n);
          For i:= 1 To m Do
               h_mat4[i]:=nil;
          Read_holmat(h_mat4,m,n);
          sum_Elements(h_mat4,m,s);
          Display(h_mat4,m,n);
          Writeln('************The  Array is ************');
          Writeln('--------------------------------------------');
          Writeln(' The  Sum Element of Array is =',s:4:2);
          Write('Press Enter To main Menu.......');
          Readln;
      End;
      '5':
      Begin
          Write('Enter the line number=');   Readln(m);
          Write('Enter the column number='); Readln(n);
          For i:= 1 To m Do
               h_mat5[i]:=nil;
          Read_holmat(h_mat5,m,n);
          greatest(h_mat5,m,g);
          Writeln('************The  Array is ************');
          Writeln('--------------------------------------------');
          Display(h_mat5,m,n);
          Writeln(' the Greatest Element in This  Array is :',g:4:2);
          Write('Press Enter To main Menu.......');
          Readln;
      End;
  End;{End Case}
until c='6';
End.
          Write('Press Enter To main Menu.......');
          Readln;
      End;
  End;{End Case}
until c='6';
End.
```

## المسألة 13:

يمكن تمثيل كثير الحدود بجدول array نخزن فيه أمثال كثير الحدود وبعددصحيح يمثل درجته، ولنضع حداً أعظم لدرجة كثير الحدود القيمة 30.

يمكن أن نعرف نمط كثير الحدود Polynom كمايلي:

```
Type  Polynm= record
Begin
Cof: array [0..30] of real degree: integer
End;
```

وهكذا لتمثيل كثير الحدود 1+ 3x10 في متحول p. (var p: polynom). نقوم بعمليات الإسناد :

```
p.degree: =10;
For i:= 0 to 10 do
  p. cov [i] :=0;
p.cov[0] := 1;
p.cov[10] := 3;
```

والأن قد استوعبنا تعريف نمط كثير الحدود، المطلوب هو كتابة وحدة برمجية unit، تتضمن تعريف النمط ومجموعة البرامج الجزئية التالية التي تتعامل معه:

Real Pol: تعريف يقرأ كثير الحدود من الدخل، يعطى الدخل بإعطاء الدرجة والمثل لكل حد مختلف عن الصفر.

WritePol: يكتب كثير الحدود، ويظهر الحد بالشكل (xxxxxx.xx)X^n للحدود غير الصفرية فقط. خذ بالحسبان كتابة كثير الحدود الطويل على أكثر من سطر.

ZeroPol: يعطي لكثير الحدود ، القيمة صفر.

AdPol: يجمع كثير الحدود، ويضع النتائج في كثير حدود آخر.

SubPol: يطرح كثير الحدود، ويضع الناتج في كثير حدود آخر.

ProdPol: يضرب كثير الحدود، ويضع التتائج في كثير حدود آخر.

اكتب برنامجاً يستخدم الوحدة البرمجية السابقة ويختبر جميع عملياتها.

```
Type
    P_polynom=^Polynom;
    Polynom=Record
      Degree:integer;
      Cof:real;
      Next:p_polynom;
    End;
Var
     ls,L1,l2,l3:p_polynom;
     C:char;
{---------------------------------------------------------------------------}
Procedure Space;
Var
   I:INTEGER;
Begin
   For I:= 1 To 100 Do
      Writeln;
End;{End Prcedure}
{----------------------------------------------------------------------------}
Procedure Menu(Var C:Char);
Begin
  Space;
  Writeln('|--------------------------------------------------------------------|');
  Writeln('|                                                                    |');
  Writeln('|             WELCOME IN FATHER AND BROTHER PROGRAM                  |');
  Writeln('|                                                                    |');
  Writeln('|     WELCOME IN FATHER AND BROTHER PROGRAM                          |');
  Writeln('|                                                                    |');
  Writeln('|                   WHAT DO YOU WANT TO DO ?                         |');
  Writeln('|                                                                    |');
  Writeln('|                    -------------------                             |');
  Writeln('|                    |    MAIN MENU    |                             |');
  Writeln('|                    -------------------                             |');
  Writeln('|                                                                    |');
  Writeln('|                                                                    |');
  Writeln('|                1: TO ADD(READ)POLYNOM.                             |');
  Writeln('|                                                                    |');
  Writeln('|                                                                    |');
  Writeln('|                2: TO PRINT POLYNOM.                                |');
  Writeln('|                                                                    |');
  Writeln('|                                                                    |');
  Writeln('|                3: TO SUM TWO POLYNOMS.                             |');
  Writeln('|                                                                    |');
  Writeln('|                                                                    |');
  Writeln('|                4: TO SUB TWO POLYNOMS.                             |');
  Writeln('|                                                                    |');
  Writeln('|                                                                    |');
```

```
  Writeln('|                 5: TO MUL TWO POLYNOMS.                          |');
  Writeln('|                                                                  |');
  Writeln('|                                                                  |');
  Writeln('|                 6: TO EXIT FROM PROGRAM.                         |');
  Writeln('|                                                                  |');
  Writeln('|------------------------------------------------------------------|');
  Writeln(' ENTER ONE OF THE FOLOWING CHOICES ');
  Readln(C);
End;
{-----------------------------------------------------------------------------}
Procedure Insert_poly(var Ls:p_polynom; poly:polynom);
Var
   p,prev,t:p_polynom;
Begin
   New(t);
   t^:=poly;
   t^.next:=Nil;
   If Ls=Nil Then
      Ls:=t
   Else {*}
   Begin
     If t^.Degree>ls^.Degree Then
       Begin
         t^.next:=ls;
         ls:=t;
        End
     Else {**}
     Begin
        p:=Ls;
        while ((p<>Nil) And (t^.Degree<p^.Degree)) Do
         Begin
           prev:=p;
           p:=p^.next;
         End;
         If  (t^.Degree<>p^.Degree) Then
           Begin
             t^.next:=p;
             prev^.next:=t;
           End
           Else
               p^.cof:=p^.cof+t^.cof;
     End;{End **}
   End;{End *}     End;{End Procedure}
End;{End Procedure}
Procedure  Read_Poly(Var p:p_polynom);
Var  R:polynom;
Begin
   Writeln(' To Start Press Enter , But To Exit press CTRL+Z ... ');
   While Not Eof Do
   Begin
      Readln;
      Write('Enter The Cof ='); Readln(R.cof);
      Write('Enter The Degree='); Readln(R.degree);
```

```
      Insert_poly(P,R);
      Writeln(' To Continue Press Enter , But To Exit press CTRL+Z ... ');
  End;
End; {End procedure}

{-----------------------------------------------------------------------------}
Procedure Mul_poly(ls1,ls2:p_polynom; Var ls3:p_polynom);
Var
    t1,t2:p_polynom;  R:polynom;
Begin
  t1:=ls1;
  While t1<>nil Do
  Begin
    t2:=ls2;
    While t2<>nil Do
    Begin
        R.cof:=((t1^.cof)*(t2^.cof));
        R.Degree:=(t1^.degree+t2^.Degree);
        Insert_poly(ls3,R);
        t2:=t2^.next;
    End;
    t1:=t1^.next;
  End;
End; {End procedure}

Procedure  Sum_poly(ls1,ls2:p_polynom; Var ls3:p_polynom);
Begin
   While  ls2<>nil do
   begin
      Insert_poly(ls3,ls2^);
      ls2:=ls2^.next;
   End;
   While ls1<>nil do
   Begin
       Insert_poly(ls3,ls1^);
       ls1:=ls1^.next;
   End;
End;{End procedure}
{-----------------------------------------------------------------------------}
Procedure  Sub_poly(ls1,ls2:p_polynom; var ls3:p_polynom);
Begin
   While  ls1<>nil do
   begin
      Insert_poly(ls3,ls1^);
      ls1:=ls1^.next;
   End;
   While ls2<>nil do
   Begin
       ls2^.cof:=((-1)*(ls2^.cof));
       Insert_poly(ls3,ls2^);
       ls2:=ls2^.next;
   End;
End;{End procedure}
```

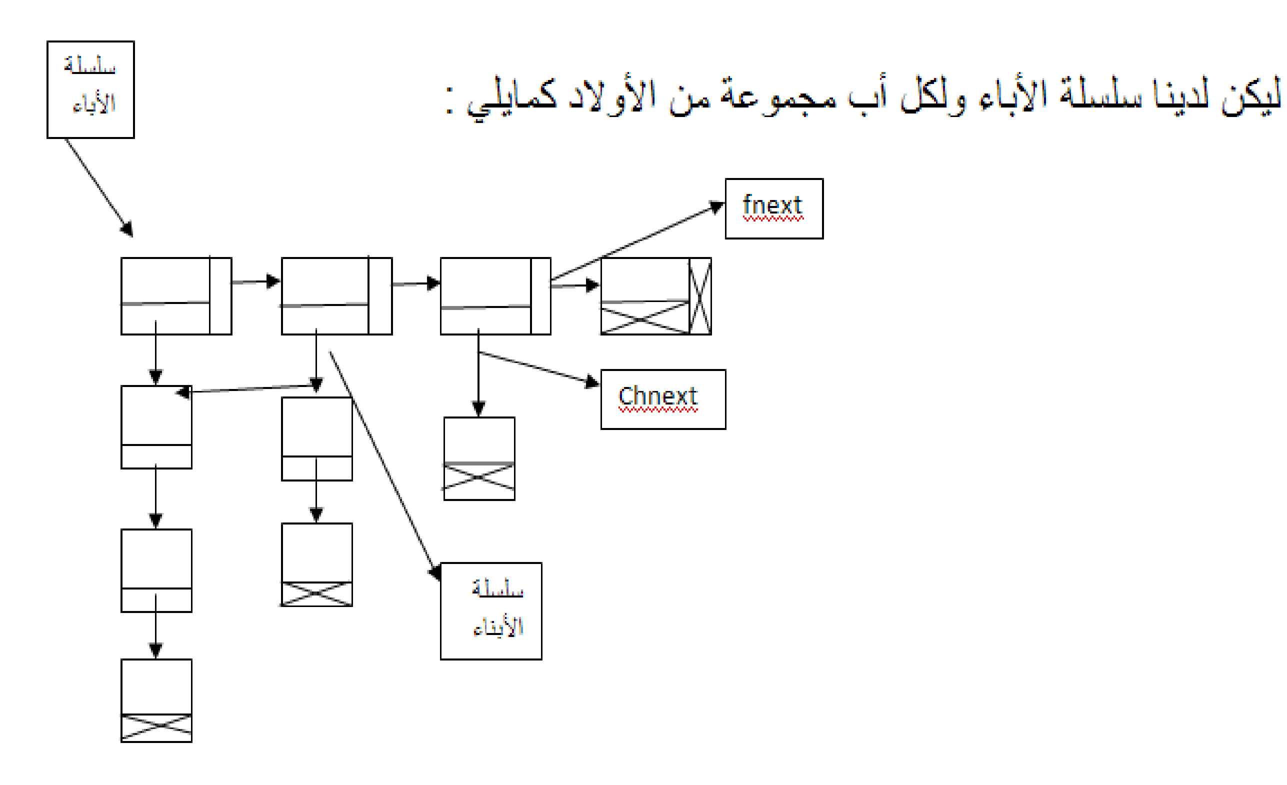
```
Procedure Write_poly( p:p_polynom);
Var
    t:p_polynom;
Begin
   t:=p;
   While t<>nil do
   Begin
     Write('(',t^.cof:4:2,'x^',t^.degree,')');
     t:=t^.next;
   End;
  Writeln;
End;{End procedure}
{-----------------------------------------------------------------------------}
   t:=p;
   While t<>nil do
   Begin
     Write('(',t^.cof:4:2,'x^',t^.degree,')');
     t:=t^.next;
   End;
  Writeln;
End;{End procedure}
Begin{*****************************Main program*****************************}
   Ls:=nil;
Repeat
   l1:=nil;
   l2:=nil;
   l3:=nil;
  Menu(c);
  If c='1' Then
  Begin
     Read_poly(Ls);
     Writeln(' Press Enter  To continue...... ');
     Readln;
  End;
  If c='2' Then
  Begin
     Write_poly(Ls);
     Writeln(' Press Enter  To continue...... ');
     Readln;
  End;
  If c='3' then
  Begin
    Writeln(' Enter The polynom  1  ');
    Writeln('----------------------');
    Read_poly(l1);
    Writeln('---------------------------------------------------------');
    Writeln(' Enter The polynom  2  ');
    Writeln('----------------------');
    Writeln(' Enter The polynom  2  ');
    Writeln('----------------------');
    Read_poly(l2);
    Write_poly(l1);
    Writeln('---------------------------------------------------------');
```

```
    Write_poly(l2);
    Writeln('---------------------------------------------------------');
    Sum_poly(l1,l2,l3);   {sum peocedure}
    Write_poly(l3);
    Writeln('---------------------------------------------------------');
    Writeln(' Press Enter  To continue...... ');
    Readln;
  End;
  If c='4' then
  Begin
    Writeln(' Enter The polynom  1  ');
    Writeln('----------------------');
    Read_poly(l1);
    Writeln('---------------------------------------------------------');
    Writeln(' Enter The polynom  2  ');
    Writeln('----------------------');
    Read_poly(l2);
    Write_poly(l1);
    Writeln('---------------------------------------------------------');
    Write_poly(l2);
    Writeln('---------------------------------------------------------');
    sub_poly(l1,l2,l3);  {sub procedure}
    Write_poly(l3);
    Writeln(' Press Enter  To continue...... ');
    Readln;
  End;
  If c='5' then
  Begin
    Writeln(' Enter The polynom  1  ');
    Writeln('----------------------');
    Read_poly(l1);
    Writeln('---------------------------------------------------------');
    Writeln(' Enter The polynom  2  ');
    Writeln('----------------------');
    Read_poly(l2);
    Write_poly(l1);
    Writeln('---------------------------------------------------------');
    Write_poly(l2);
    Writeln('---------------------------------------------------------');
    mul_poly(l1,l2,l3);   {mul peocedure}
    Write_poly(l3);
    Writeln('---------------------------------------------------------');
    Writeln(' Press Enter  To continue...... ');
    Readln;
  End;
    Writeln(' Enter The polynom  1  ');
    Writeln('----------------------');
    Read_poly(l1);
    Writeln('---------------------------------------------------------');
    Writeln(' Enter The polynom  2  ');
    Writeln('----------------------');
    Read_poly(l2);
    Write_poly(l1);
```

```
    Writeln('---------------------------------------------------------');
    Write_poly(l2);
    Writeln('---------------------------------------------------------');
    mul_poly(l1,l2,l3);   {mul peocedure}
    Write_poly(l3);
    Writeln('---------------------------------------------------------');
    Writeln(' Press Enter  To continue...... ');
    Readln;
  End;
until c='6';
End.
```

# المسألة 14 : مسألة الأباء والأبناء

ليكن لدينا سلسلة الأباء ولكل أب مجموعة من الأولاد كمايلي :




بنية المعطيات:

```
TYPE
  P_CHILD=^CHILD;
  CHILD=RECORD
     CNAME:STRING[20];
     CNEXT:P_CHILD;
  END;
  P_FATHER=^FATHER;
  FATHER=RECORD
     FNAME:STRING[20];
     FNEXT:P_FATHER;
     CHNEXT:P_CHILD;
  END;
```

والمطلوب:

1- إضافة أب إلى سلسة الأباء في مكانه الصحيح ضمن سلسلسة الأباء .
2- إضافة أبن إلى أب موجود ضمن سلسلة الأباء .
3- حذف أبن لأب موجود ضمن سلسلة الأباء.
4- حذف أب بالنسبة لأبن موجود ضمن سلسلة الأبناء.
5- طباعة أبناء أب معين موجود ضمن سلسلة الأباء.
6- طباعة أسماء جميع الأباء الذين ليس لديهم أولاد.
7- طباعة كل الأباء.
8- طباعة أسم الاب الذي لدية أكبر عدد من الأولاد.

```
 TYPE
  P_CHILD=^CHILD;
  CHILD=RECORD
     CNAME:STRING[20];
     CNEXT:P_CHILD;
  END;
  P_FATHER=^FATHER;
  FATHER=RECORD
     FNAME:STRING[20];
     FNEXT:P_FATHER;
     CHNEXT:P_CHILD;
  END;
VAR
    S,SON:STRING;
    L_S:P_FATHER;
    I:INTEGER;
    C,C1:CHAR;
    F:BOOLEAN;
{------------------------------------------------------------------------------}
PROCEDURE SPACE;
VAR I:INTEGER;
BEGIN
FOR I:= 1 TO 100 DO
WRITELN;
END;{END PROCEDURE}
{------------------------------------------------------------------------------}
PROCEDURE MENU(VAR C:CHAR);
BEGIN
SPACE;
WRITELN('|------------------------------------------------------------------------------|');
WRITELN('|                                                                              |');
WRITELN('|             WELCOME IN FATHER AND BROTHER PROGRAM                            |');
WRITELN('|                                                                              |');
WRITELN('|                   WHAT DO YOU WANT TO DO ?                                   |');
WRITELN('|                    -------------------                                       |');
WRITELN('|                    |    MAIN MENU    |                                       |');
WRITELN('|                    -------------------                                       |');
WRITELN('|                                                                              |');
WRITELN('|    1: TO ADD FATHER INTO SERIES IN THE CORRECT PLACE.                        |');
WRITELN('|                                                                              |');
WRITELN('|                                                                              |');
WRITELN('|    2: TO ADD CHILD ACORDING TO  FATHER  FOUND.                               |');
WRITELN('|                                                                              |');
WRITELN('|                                                                              |');
WRITELN('|    3: TO DELETE SON FOR FATHER THAT FOUND IN SERIES.                         |');
WRITELN('|                                                                              |');
WRITELN('|    4: TO DELETE FATHER FOR SON THAT FOUND IN SERIES.                         |');
WRITELN('|                                                                              |');
WRITELN('|                                                                              |');
WRITELN('|    5: TO PRINT  CHILDREN FOR CERTION FATHER.                                 |');
WRITELN('|                                                                              |');
WRITELN('|                                                                              |');
WRITELN('|    6: TO PRINT ALL NAMES OF FATHERS THAT NO CHILDREN.                        |');
```

```
WRITELN('|                                                                              |');
WRITELN('|                                                                             |');
WRITELN('|    7: TO PRINT ALL FATHERS.                                                 |');
WRITELN('|                                                                             |');
WRITELN('|                                                                             |');
WRITELN('|    8: TO PRINT FATHER NAME WHO HAS GREATEST NUMBER OF CHILDREN.             |');
WRITELN('|                                                                             |');
WRITELN('|    9: TO EXIT FROM PROGRAM.                                                 |');
WRITELN('|-----------------------------------------------------------------------------|');
WRITE(' ENTER ONE OF THE FOLOWING CHOICES ');
READLN(C);
END;
{-----------------------------------------------------------------------------}
PROCEDURE  D_F(VAR LS:P_FATHER; FN:STRING);
VAR T,TE:P_FATHER;
   DEL,FOUND:BOOLEAN;
BEGIN
    DEL:=FALSE;     FOUND:=FALSE;
   WHILE ((LS<>NIL) AND (NOT DEL)) DO
   BEGIN
       IF FN=LS^.FNAME THEN
         BEGIN
            DEL:=TRUE;
            T:=LS;                  {DELETE FATHER 4}
            LS:=LS^.FNEXT;
            DISPOSE(T);
         END
       ELSE
         BEGIN
            TE:=LS;        FOUND:=FALSE;
            WHILE ((TE^.FNEXT<>NIL) AND (NOT FOUND))DO
             BEGIN
               IF TE^.FNEXT^.FNAME=FN THEN
                  FOUND:=TRUE
               ELSE
                  TE:=TE^.FNEXT;
             END;
             IF FOUND=FALSE THEN
                  WRITELN(' FATHER NOT FOUND ......')
             ELSE
                  BEGIN
                       T:=TE^.FNEXT;
                       TE:=TE^.FNEXT^.FNEXT;
                       DISPOSE(T);
                       DEL:=TRUE;
                  END;
         END;
   END;
END;{END PROCEDURE}
{-----------------------------------------------------------------------------}
         END;
   END;
END;{END PROCEDURE}
```

```
{----------------------------------------------------------------------------}

FUNCTION  SEARCH(FN:STRING; VAR LS:P_FATHER;VAR FOUND:BOOLEAN):P_FATHER;
VAR
    T:P_FATHER;
                  {SERCHING FOR FATHER NAME IN THE SERIES}
BEGIN
  T:=LS;
  FOUND:=FALSE;
  WHILE ((T<>NIL) AND (NOT FOUND)) DO
  BEGIN
    IF FN=T^.FNAME THEN
        FOUND:=TRUE
    ELSE
       T:=T^.FNEXT;
  END;{END WHILE}
  SEARCH:=T;
END;{END FUNCTION}
{----------------------------------------------------------------------------}
PROCEDURE  SEARCH_SON(LS:P_FATHER; SON:STRING;VAR P:P_CHILD; VAR FOUND:BOOLEAN);
VAR      {THIS FUNCTION THAT GIVE IT POINTER FOR FATHER AND SEARCH FOR SON
              AND RETURN VALUE IF BE (SON NAME FOUND IN THE FATHER NAME) }
    T:P_CHILD;
BEGIN
      P:=LS^.CHNEXT;  FOUND:=FALSE;
     WHILE  ((P<>NIL) AND (NOT FOUND)) DO
     BEGIN
       IF P^.CNAME=SON THEN
            FOUND:=TRUE
       ELSE
            P:=P^.CNEXT;
   END;
P:=LS^.CHNEXT;
END;{END FUNCTION}

{----------------------------------------------------------------------------}
PROCEDURE INSERT_FATHER(VAR LS:P_FATHER; S:STRING);
VAR
   TEMP,P,PREV:P_FATHER;
BEGIN                {ORDER (1) ADD FATHER TO SERIES}
  NEW(TEMP);
  TEMP^.FNAME:=S;
  TEMP^.FNEXT:=NIL;   {PFATHER: POINTER FOR NEXT FATHER}
  TEMP^.CHNEXT:=NIL;    {CHNEXT:POINTER FOR HER CHILDRENS}
  IF LS=NIL THEN
     LS:=TEMP
  ELSE {*}
      BEGIN
         IF S<LS^.FNAME THEN
             BEGIN                  {1}
               TEMP^.FNEXT:=LS;
               LS:=TEMP;
             END
```

```
         ELSE {**}
            BEGIN
               P:=LS;
               PREV:=P;
               WHILE ((P<>NIL) AND (TEMP^.FNAME>P^.FNAME)) DO
               BEGIN
                 PREV:=P;
                 P:=P^.FNEXT;
               END;{END WHILE}
               IF (TEMP^.FNAME<>P^.FNAME) THEN
                BEGIN
                  TEMP^.FNEXT:=P;
                  PREV^.FNEXT:=TEMP;
                END;{END IF}
             END;{END ELSE(**)}
      END;{END ELSE(*)}
END;{END PROCEDURE}
{-----------------------------------------------------------------------------}
                  PREV^.FNEXT:=TEMP;
                END;{END IF}
             END;{END ELSE(**)}
      END;{END ELSE(*)}
END;{END PROCEDURE}
{-----------------------------------------------------------------------------}
PROCEDURE  INSERT_CHILD(VAR LS:P_FATHER; FN,SON:STRING);
VAR  T:P_FATHER;
     T1,P,PREV:P_CHILD;
     F:BOOLEAN;
BEGIN
  T:=LS;
  NEW(T1);                          {2}
  T1^.CNAME:=SON;
  T1^.CNEXT:=NIL;
  T:=SEARCH(FN,LS,F);
  IF F=FALSE THEN
      WRITELN(' THE FATHER NAME IS NOT FOUND ') {FATHER NOT FOUND}
  ELSE {* IF FATHER NAME FOUND (FOUND=TRUE)}
       BEGIN
            IF T^.CHNEXT=NIL THEN    {IF THE SERIES IS EMPTY}
               T^.CHNEXT:=T1   {POINTER FOR FATHER IS PINT FOR CHILDREN}
            ELSE {** IF THE SERIES NOT EMPTY}
            BEGIN
                 IF  SON<T^.CHNEXT^.CNAME THEN
                 BEGIN      {IF  THE SON NAME IS SMALER THAN FIRST ELEMENT}
                      T1^.CNEXT:=T^.CHNEXT;
                      T^.CHNEXT:=T1;
                 END{END WHILE}
                 ELSE {*** ANY WHERE}
                 BEGIN
                      P:=T^.CHNEXT;
                      PREV:=P;
                      WHILE ((P<>NIL) AND (SON>P^.CNAME)) DO
                      BEGIN
```

```
                              PREV:=P;
                              P:=P^.CNEXT;
                         END;{END WHILE}
                         IF SON<>T^.CHNEXT^.CNAME THEN
                         BEGIN
                              T1^.CNEXT:=P;
                              PREV^.CNEXT:=T1;
                         END;{EN IF}
                    END;{END ELSE ***}
               END;{END ELSE **}
   END;{END ELSE *}
END;{END PROCEDURE}
{------------------------------------------------------------------------------}
PROCEDURE DELETE_SON(FN,SON:STRING; VAR LS:P_FATHER);
VAR  T:P_FATHER;
     P,PREV,TEMP,NEXT,p1:P_CHILD;
     FOUND:BOOLEAN;
BEGIN                                            {3}
    T:=SEARCH(FN,LS,FOUND);
    IF FOUND=FALSE THEN
         WRITELN(' THE FATHER NAME IS NOT FOUND .....')
    ELSE{1}
    BEGIN
        P:=T^.CHNEXT;
        IF P=NIL THEN
            WRITELN(' THIS FATHER  DOES NOT HAS CHILDREN.....')
        ELSE {2}  {IF NOT EMPTY}
          BEGIN
            IF P^.CNAME=SON THEN  {P=T^.CHNEXT }
             BEGIN
               tEMP:=P;
               P:=P^.CNEXT;        {DELETE FIRST ELEMENT}
               DISPOSE(TEMP);
               T^.CHNEXT:=P; {Very important To Move Ls(T^.chnext to down(p))}
               P:=P^.CNEXT;        {DELETE FIRST ELEMENT}
               DISPOSE(TEMP);
               T^.CHNEXT:=P; {Very important To Move Ls(T^.chnext to down(p))}
              END
             ELSE {3}
               BEGIN    {ANY WHERE}
                   NEXT:=P;  FOUND:=FALSE;
                   WHILE ((P<>NIL)AND(NOT FOUND))  DO
                    BEGIN
                       IF P^.CNEXT^.CNAME<>SON THEN
                           P:=P^.CNEXT
                       ELSE
                           FOUND:=TRUE;
                    END;{END WHILE}
                    {POINTER STOP BEFOR THE ELEMENT THAT YOU WANT TO DELETE }
                    IF FOUND=FALSE THEN
                        WRITELN('THE NAME OF CHILD IS NOT FOUND ....')
                    ELSE {4}
                      BEGIN
```

```
                         TEMP:=P^.CNEXT;
                         P^.CNEXT:=P^.CNEXT^.CNEXT; {OR P^.CNEXT:=TEMP^.CNEXT}
                         DISPOSE(TEMP);
                      END;{EN ELSE 4}
               END;{END ELSE 3}
        END;{END ELSE 2}
    END;{END ELSE 1}
END;
{-----------------------------------------------------------------------------}
PROCEDURE DELETE_F(SON:STRING; VAR LS:P_FATHER);
VAR
   T:P_FATHER;
   TEMP,P,TE:P_CHILD;
   F:BOOLEAN;
   FN:STRING;
BEGIN
     F:=FALSE;   T:=LS;
     WHILE  ((T<>NIL) AND (NOT F)) DO
     BEGIN
         SEARCH_SON(T,SON,P,F);
         IF  F=TRUE THEN
            TE:=P
         ELSE
            T:=T^.FNEXT;
     END;
{THE LOOP THAT STOP AND(TE:P_CHILD (TE: IS LIST START FOR FATHER SERIES)}
     IF F=FALSE THEN
        WRITELN(' THE  FATHER NAME IS  NOT FOUND ')
     ELSE
     BEGIN
      FN:=T^.FNAME;
      WHILE  TE<>NIL DO  { TE=P^.CHNEXT ; LIST START FOR FATHER SERIES}
      BEGIN
        TEMP:=TE;
        TE:=TE^.CNEXT;
        DISPOSE(TEMP);
        T^.CHNEXT:=TE;
      END;{END WHILE}   {T: POINTER FOR FATHERS}
        D_F(LS,FN);
    END;
{IF F=TRUE THEN  }
END;{END PROCEDURE}
{-----------------------------------------------------------------------------}
    P1,P2:P_CHILD;
    I,E:INTEGER;
    FN:STRING;
BEGIN
      T:=LS;  E:=0;
      WHILE T<>NIL DO
      BEGIN
         P1:=T^.CHNEXT;  I:=0;
         WHILE  P1<>NIL DO
           BEGIN
```

```
            I:=I+1;
            P1:=P1^.CNEXT;
          END;
        IF I>E THEN
        BEGIN
          E:=I;
          FN:=T^.FNAME;
        END;
        T:=T^.FNEXT;
   END;
WRITELN(FN);
END; {END PROCEDURE}
PROCEDURE VIEW(LS:P_FATHER);
VAR T:P_FATHER;
BEGIN
  T:=LS;
  WHILE T<>NIL DO
   BEGIN
    IF T^.CHNEXT=NIL THEN        {6}
         WRITELN(T^.FNAME);
    T:=T^.FNEXT;
   END;{END WHILE}
END;{END PROCEDURE}
PROCEDURE VIEW_FATHER(LS:P_FATHER);
VAR T:P_FATHER;
BEGIN
  T:=LS;
  WHILE T<>NIL DO
   BEGIN                      {7}
         WRITELN(T^.FNAME);
         T:=T^.FNEXT;
   END;{END WHILE}
END;{END PROCEDURE}
{---------------------------------------------------------------------------}
PROCEDURE V_C(FN:STRING; VAR LS:P_FATHER);
VAR
  F:BOOLEAN ;
  TE   :P_FATHER;
  C    :P_CHILD ;
BEGIN                              {5}
     TE:=SEARCH(FN,LS,F);
     IF F=FALSE THEN
        WRITELN(' THE FATHER NAME IS NOT FOUND ')
     ELSE
     BEGIN
          C := TE^.CHNEXT ;
          WHILE (C<>NIL) DO
          BEGIN
               WRITELN(C^.CNAME);
               C:=C^.CNEXT;
          END; {END WHILE}
     END;{END ELSE}
END;{END PROCEDURE}
```

```
{--------------------------------------------------------------------------}
BEGIN{*****************************MAIN PROGRAM*****************************}
L_S:=NIL;
REPEAT
L_S:=NIL;
REPEAT
MENU(C);
IF C='1' THEN
BEGIN                    {1 ADD FATHER }     {*****READY****}
  WRITE('ENTER NAME OF FATHER: '); READLN(S);
  INSERT_FATHER(L_S,S);
  WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
  READLN;
END;
IF C='2' THEN
BEGIN                 { 2  ADD SON}
   WRITE('ENTER NAME OF FATHER: '); READLN(S);   {*****READY****}
   WRITE('ENTER NAME OF SON '); READLN(SON);
   INSERT_CHILD(L_S,S,SON);
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
   READLN;
END;
IF C='3' THEN
BEGIN
   WRITE('ENTER NAME OF FATHER:'); READLN(S);
   WRITE('ENTER NAME OF CHILD:');  READLN(SON);
   DELETE_SON(S,SON,L_S);
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
   READLN;
END;
IF C='4' THEN
BEGIN
WRITE('ENTER THE NAME OF SON:'); READLN(S);
DELETE_F(S,L_S);
V_C(S,L_S);
END;
IF C='5' THEN
BEGIN
   WRITE('ENTER THE NAME OF FATHER :'); READLN(S);
   V_C(S,L_S);                             {*****READY****}
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
   READLN;
END;
IF C='6' THEN
BEGIN
   VIEW(L_S);
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
   READLN;
END;
IF C='7' THEN
BEGIN              {7}
   VIEW_FATHER(L_S);
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
```

```
   READLN;
END;
IF C='8' THEN
BEGIN
   GRATEST_C(L_S);
   WRITE(' PRESS ANY KEY TO MAIN MENU.... ');
   READLN;
END;
IF C='9' THEN
BEGIN
WRITE('ARE YOU SURE TO EXIT FROM PROGRAM,TYPE Y/N ? '); READLN(C1);
END;
UNTIL ((C='9') AND (C1 IN['Y','y']));
END.
```